Versatile IPv6 Mobility Deployment with Dual Stack Mobile IPv6

Romain Kuntz
LSIIT (UMR CNRS 7005)
Louis Pasteur University Strasbourg, France
kuntz@lsiit.u-strasbg.fr

Jean Lorchat
Internet Initiative Japan Inc.,
Tokyo, Japan
jean@iij.ad.jp

ABSTRACT

In this paper, we show how *Mobile IPv6 support for dual stack Hosts and Routers* (DSMIPv6) can be used as a very efficient, automatic and autonomous way of network planning as well as a multipurpose mobility solution. Along with the details of the Linux-based implementation, we show the results of measurements made against a test scenario that heavily relies on DSMIPv6. We also discuss the next steps and remaining specification issues that should be tackled in order to ensure a rapid deployment of the protocol.

Categories and Subject Descriptors

C.2.2 [**Network Protocols**]: Protocol verification

General Terms

Design, Experimentation, Performance

Keywords

IPv6, Mobility, DSMIPv6, Implementation

1. INTRODUCTION

Current trends observed worldwide with respect to Internet growth show that the number of connected users is growing very fast. And while the IPv6 deployment is slowly taking off, there can be a present need for advanced features promised by next generation protocols.

Especially, in the same way as Network Address Translation (NAT) allows a host to provide global connectivity to a whole network albeit with restrictions, NEtwork MObility Basic Support (NEMO BS [1]) allows a router with a single address to provide mobility agnostic end-to-end access to all attached nodes. This feature of NEMO BS makes it very suitable for many scenarios around public transportation [2].

In NEMO BS, the mobility management operations are handled by the mobile network router known as the Mobile Router (MR). Legacy IPv6 nodes (Mobile Network Nodes or MNNs) located in the mobile network are provided with a global IPv6 address which remains valid whatever the location of the mobile network. While moving from one access network to another, the MR maintains an IPv6-in-IPv6 tunnel with a remote fixed node called the Home Agent (HA). Both the MR and the HA maintain a relation between a permanent IPv6 address assigned to the MR (called the Home Address or HoA) and a temporary address (the Care-of Address or CoA) that the MR gets along with its movement in each foreign network. This relation is maintained through the usage of the Binding Update (BU) and Binding Acknowledgment (BAck) messages regularly exchanged between the MR and its HA. All network movements are therefore transparent to the nodes located in the mobile network, the MR and the HA performing the necessary operations to route packets destined to or originated from the mobile network.

However, in its current state, NEMO BS is restricted to pure IPv6 operation, whereas only few modifications would allow it to operate using an IPv4 CoA directly. This is under standardization at the IETF under the name *Mobile IPv6 support for dual stack Hosts and Routers* or DSMIPv6 [3]. Using an IPv4 CoA allows many pioneering scenarios like the rapid deployment of a whole network in emergency situations, with very few requirements apart from a basic IPv4 access, even using NAT, because DSMIPv6 specifies a NAT traversal mechanism. In addition, from the public transportation scenarios, it makes the in-vehicle routers less complex by removing the tunneling layer required when visiting IPv4-only networks. DSMIPv6 is thus a very suitable architecture for the future IPv6 mobility deployment.

This paper is organized as follow: after an overview of dual stack mobility in the literature, we outline in section 2 two environments where its usage would play a leading role. We then detail in section 3 our DSMIPv6 implementation. An evaluation presented in section 4 endorses the feasibility of our scenario. We then review the future work in section 5, and finally expose in section 6 possible issues that could refrain from adopting DSMIPv6.

2. DUAL STACK MOBILITY

2.1 State of the Art

Supporting dual stack mobility by addressing IPv4 and IPv6 mobility separately could result in an inefficient mobility management as explained in [4]. For example, operating both Mobile IPv4 [5] and Mobile IPv6 [6] on a single node would require that both IPv6 and IPv4 are operated in each of the visited network while the Mobile Node (being a host or a router) is moving in the Internet. For example, a node connecting to an IPv6-only network would not be able to operate Mobile IPv4 anymore, thus disrupting all of its IPv4 applications. Furthermore, managing two different protocols creates operational overhead both on the Mobile Node (MN) and

MobiArch'08, August 22, 2008, Seattle, Washington, USA.

the network as it mandates to operate the necessary software and infrastructure on the MN and at the operator level.

A first solution [7] combines IPv6-in-IPv4 tunnelling with NAT-PT [8] to manage handovers of a MN operating Mobile IPv6 in IPv4-only networks. Tunnelling provides IPv6 connectivity to the node in the IPv4 network, while NAT-PT takes care of translating the IPv6 packets into IPv4 (and conversely) when the node communicates with an IPv4-only correspondent. One major operational overhead of this proposal is that all tunnelled packets are extracted to be translated by a NAT-PT device that must be located on the path between the communicating peers. Furthermore, this solution addresses IPv6 mobility in IPv4 networks, but not IPv4 mobility at all. How to handle NAT in the visited IPv4 networks is also not considered.

Another proposal [9] defines a small extension to Mobile IPv4 that enables the registration of an IPv6 address to the HA while the MN is in an IPv4 network. IPv6 packets originated from or destined to the MN are encapsulated in an IPv4 header between the MN and its HA. This solution is only designed to provide IPv6 connectivity in IPv4 networks at the cost of an extra IP encapsulation, and does not address the possibility to roam in pure IPv6 networks. This work served as a base to define *Dual Stack Mobile IPv4* (DSMIPv4 [10]) which also allows a node to use IPv4 and IPv6 HoAs. However, this solution relies on the Mobile IPv4 signaling, thus preventing the node from roaming in IPv6-only networks.

The work presented in [11] explains how Mobile IPv6 and Mobile IPv4 can be operated at the same time to achieve seamless IPv6 handovers in IPv4 networks. When roaming in IPv4 networks, the Mobile IPv4 HoA is translated to a 6to4 address used as a CoA to register to the HA. All the IPv6 traffic originated from or destined to the MN then transits through the tunnel operated by Mobile IPv4. This solution results in a large header overhead due to the three encapsulations needed to transport the IPv6 packets from the node to the Mobile IPv6 HA. Furthermore, this solution does not consider the continuity of the IPv4 service when roaming in IPv6 networks.

None of the solutions described so far considers the continuity of both IPv6 and IPv4 sessions whatever the IP version in the network where the MN roams in is operated. The *Mobile IPv6 support for dual stack Hosts and Routers* specification (DSMIPv6 [3]) presented in the next section is currently discussed at the IETF[1] as a single protocol that ensures a very flexible management of both IPv4 and IPv6 mobility in either IPv4-only or IPv6-only or dual stack networks.

2.2 Dual Stack Additions to Mobile IPv6

DSMIPv6 [3] extends the Mobile IPv6 [6] and NEMO Basic Support [1] standards to allow MNs to roam in both IPv6 and IPv4-only networks. For that purpose, it defines several interesting features:

- The MN can register an IPv4 CoA to its HA and thus roam in IPv4-only networks by the use of IPv6-in-IPv4 tunnels between the MN and its HA. One consequence is the reduction of the tunneling level, as a legacy Mobile IPv6 MN would have to use an IPv6-in-IPv6-in-IPv4 tunnel.

- The MN can use an IPv4 HoA to be used with its applications when communicating with IPv4-only correspondents. The MN can thus get rid of the use of a translator, and benefit from end-to-end IPv4 communication.

[1] http://www.ietf.org

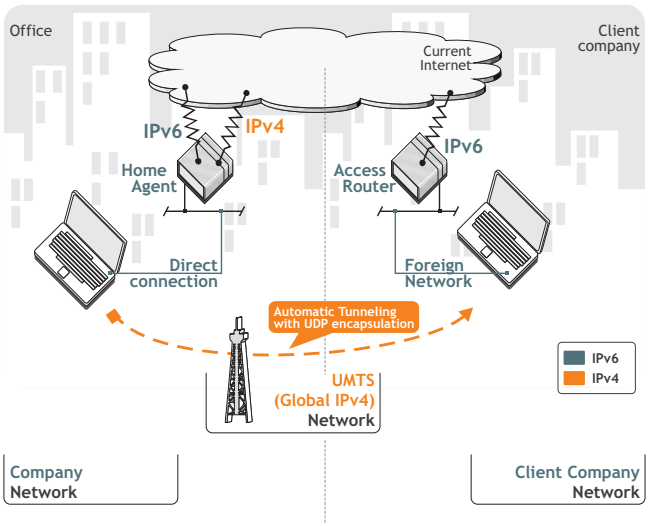

Figure 1: Transparent access between different IP networks

- A NAT detection and traversal mechanism allows the MN to communicate with its HA even though it uses an IPv4 private address as a CoA. When the MN is located behind a NAT, signaling and data are encapsulated in UDP and IPv4.

- In the case of network mobility, the MR can request an IPv4 prefix to be advertised in its mobile network, hence providing IPv4 connectivity to its MNNs, even though the mobile network is roaming in an IPv6-only network.

Thanks to these enhancements, we can imagine many possible scenarios where the use of IPv4 applications are made possible in IPv6-only networks, and conversely.

2.3 Use Cases

Previously, NEMO has attracted lots of attention as a protocol for dealing with emergency situations, from rescue-teams [12] to post-disaster infrastructure recovery [13] and easy deployments [14]. Thanks to DSMIPv6, we can move those concepts forward and realize such scenarios in heterogeneous networks.

We can assume that the full IPv6 deployment will take some time and that IPv4 will be the preferred solution in the next few years to provide Internet connectivity, especially to the end-users. We can imagine a company employee equipped with a Personal Area Network (PAN, a small network composed of embedded devices like mobile phones, PDAs, etc.). This PAN is connected to the Internet through a MR running DSMIPv6 with the NEMO BS protocol, thus taking care of the mobility management for all the devices located inside the PAN. While working in his office, the PAN owner can use the legacy NEMO BS features to roam within the office's IPv6 networks. When moving to a client company, the PAN may use NEMO BS to create a VPN with the company network, and benefit from the DSMIPv6 features to automatically roam from the IPv6 company network to the IPv4 network available through an UMTS subscription. When reaching the client company network, the PAN can transparently move again to a pure IPv6 infrastructure transparently for all the PAN devices. This use case is depicted in figure 1.

As presented in [13], a fail-safe mobile environment can be achieved by using multiple Care-of Addresses (MCoA [15]) on a node. In most of the cases, the use of a fallback CoA implies to

be multi-homed. With DSMIPv6 however, fail-over can be realized using the same access technology by registering a CoA from a different IP protocol. As an example in a mobile network environment, a transportation company may offer stable IPv6 and IPv4 connectivity to its passengers, while roaming in dual stack access networks. Even though a failure may occur at the layer 3 (for example the IPv6 access router in the roaming network is not available anymore), the MR may recover by using its IPv4 CoA and thus provide uninterrupted access to the nodes inside the moving network.

As we can see, DSMIPv6 allows to benefit from mobility in a very heterogeneous environment, and to improve the performance and overall connectivity of a network.

3. IMPLEMENTATION DETAILS

3.1 UMIP

UMIP is a set of patches for the MIPL2 software suite developed by the USAGI Project[2]. MIPL2 (Mobile IPv6 for Linux[3]) is an open-source implementation of the Mobile IPv6 standard for the GNU/Linux operating systems. UMIP aims at providing the necessary changes to MIPL2 in order to run on the latest kernels while improving the software to respect the standards.

As of May 2008, our DSMIPv6 implementation[4] is based on UMIP-0.4 with the NEPL extension[5] that enables the operation of the NEMO BS protocol. It runs on a 2.6.24 Linux kernel, one of the latest stable kernel available on GNU/Linux. The figure 2 roughly presents the current design of our DSMIPv6 extensions for UMIP. The implementation extends both the kernel and userland code, as detailed in the next sections.

3.2 XFRM extensions

XFRM [16] is a packet transformation framework residing in the Linux kernel. It allows to perform operations on IP packets such as inserting or modifying headers. This framework is for example used by UMIP to insert Destination Option and Routing Headers in the Mobile IPv6 signaling messages (BU, BAck, etc.) and add the necessary headers for IPsec.

The DSMIPv6 specification defines a NAT detection and traversal mechanism based on UDP encapsulation of the signaling and data packets. The best way to perform the UDP encapsulation is to extend XFRM to serialize this operation after all other transformations. This is especially true because IPsec transformations are already defined as XFRM transformations, which would make a userland implementation of UDP encapsulation more difficult. We thus defined an additional XFRM transformation that takes advantage of the existing framework and defines a simple UDP encapsulation scheme. This addition is split into two parts for both the reception and transmission cases.

For the transmission part, packets matching specific requirements (e.g. the BU or even data packets in case of NAT traversal) are handled by adding extra room in front of the packet. This room is then filled with necessary IPv4 and UDP headers so that the previous packet is really the payload of the new UDP packet. This new packet is then handled to the UDP network code that is going to perform destination lookup and proceed with the transmission.

As for the reception part, packets received on the DSMIPv6 UDP port that is defined in the specification (currently to be assigned by IANA) get processed by the transformation input routine. The IPv4 and UDP headers are separated from the payload which turns into a new IPv6 packet. At that point, NAT detection is started by copying the outer IPv4 source address of the packet to a dedicated field inside the packet buffer of the newly built IPv6 packet. This new IPv6 packet is then re-submitted to the network stack as a regular IPv6 packet, except that it has NAT detection information attached. From then on, the packet follows the same path as regular packets that would have come from a native IPv6 network.

The NAT information is then available for processing within the userland. It is going to be used for comparison with the contents of the IPv4 CoA option inside the BU message on the HA. If both addresses differ, it means that IPv4 source address has been rewritten on the path by a NAT device.

3.3 Userland design

The UMIP userland takes care of the movement detection, the binding management, the signaling and error processing. It also interacts with the kernel to create IP tunnels, manage the routes towards those tunnels, and add the necessary XFRM policies and states for IPsec and signaling messages transformation. Details about the implementations are described below:

- The movement detection module has been extended to detect roaming in IPv4 networks on the MN. A light DHCP client based on uDHCP[6] and triggered by a DNA [17] module (or when booting in a IPv4-only network) allows to get an IPv4 CoA when located in an IPv4 access network. The movement decision algorithm has to take into account the fact that IPv6 CoAs always have the priority upon IPv4 ones. This could be made possible by a separation of DHCP states and interface configuration. Whenever a lease is obtained, the IPv4 CoA should not be set on the interface unless no IPv6 router has been found on the link.

- The IPv4 CoA is stored in a IPv6-mapped address format and thus requires minimum changes to the data structures (such as the Binding Update List or the Binding Cache entries).

- Upon an IPv4 movement event on the MN, the userland installs XFRM policies and states in the kernel for UDP encapsulation of the BU. The HA has similar rules to decapsulate the packet and perform NAT Detection.

- When the MN is roaming in an IPv4-only network, two encapsulation methods are considered. If no NAT was detected on the path, the MN creates with its HA an IPv6-in-IPv4 SIT tunnel to encapsulate IPv6 data packets in IPv4, and maintain a default route towards this tunnel. If a NAT has been detected, new XFRM policies and states are installed by both the MN and the HA to encapsulate and decapsulate the IPv6 data traffic in IPv4 UDP.

- Security being also one of our concern, the implementation already supports the protection of the Mobile IPv6 signaling by installing the XFRM policies and states that perform IPsec operations on the BU and BAck messages. Performing NAT detection on UDP-encapsulated BU protected by IPsec is quite challenging though, as discussed later in section 6.

Our implementation does not modify the behavior of the legacy Mobile IPv6 or NEMO BS operations, and can be activated or deactivated with an option in the userland's configuration file. Even with DSMIPv6 enabled, the MN behavior is not altered as long as foreign networks can provide an IPv6 CoA to the MN.

[2]http://www.linux-ipv6.org
[3]http://www.mobile-ipv6.org
[4]http://software.nautilus6.org/DSMIP/
[5]http://software.nautilus6.org/NEPL/
[6]http://udhcp.busybox.net

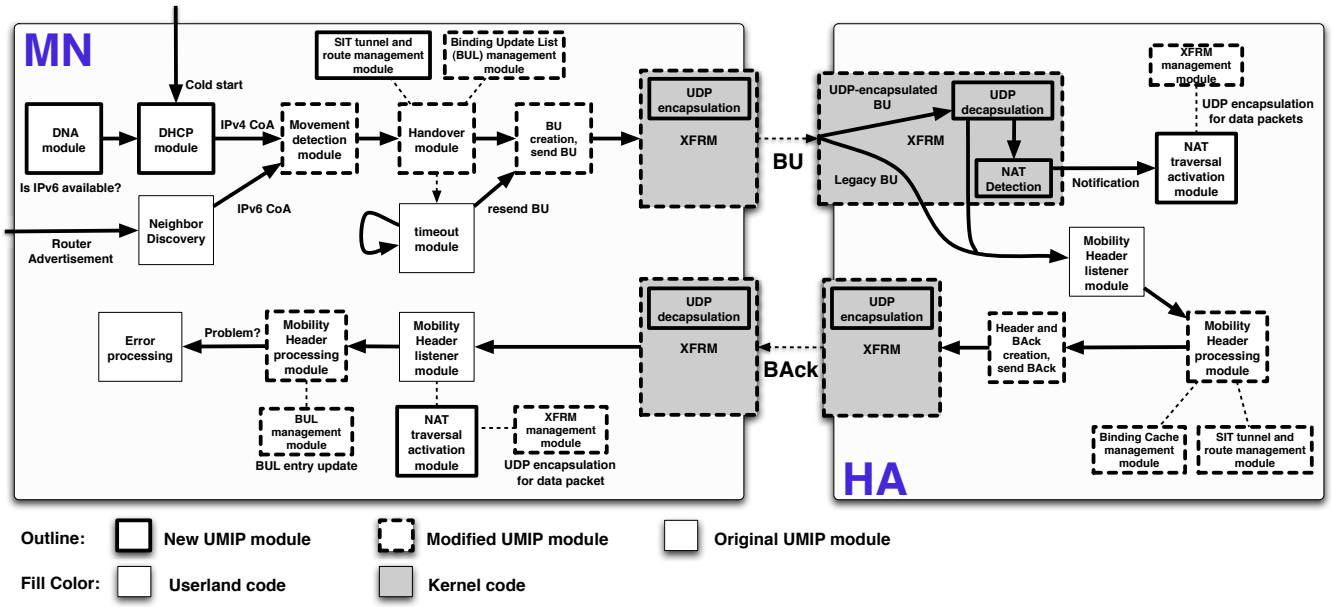

Figure 2: DSMIPv6 additions to UMIP

4. EVALUATION

4.1 Experiment Setup

In order to validate the implementation, we led some handover tests following a scenario that is equivalent to the PAN use case described in section 2.3. In this section, we show the result of such an experiment where a mobile network moves from an IPv6 network to a network providing exclusively IPv4 access and conversely. The network topology built for this experiment is presented in figure 3.

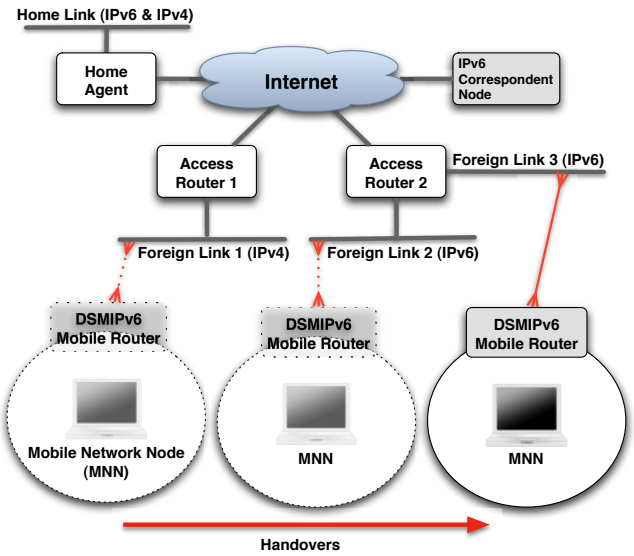

Figure 3: The experimental network

During the experiment, a MNN located inside the mobile network continuously sends an UDP flow to an IPv6 correspondent node located in the Internet (packets have an average size of 250 bytes and are sent every 5ms). We can thus analyze the MNN disconnection time experienced at the correspondent node. Also, network traces for all active interfaces are available on the MR. This allows us to benchmark DSMIPv6 operations and profile the efficiency of the various modules.

4.2 Results

4.2.1 Horizontal Handovers

A first experiment was conducted using a MR with a single egress interface, moving from an IPv6-only to an IPv4-only network, and conversely. As a comparison, a handover between two IPv6 networks is also performed. Results are shown on a 9 seconds window in figure 4. The data packet sequence number is plotted with respect to the experiment time on the left axis. The signaling packet type is plotted with respect to the experiment time on the right axis.

The handover performed between two IPv6 networks and depicted on the top graph gives approximatively the same result as the one conducted during the NEPL evaluation exposed in [18].

Link up	IPv4 config (DHCP)	MIPv6
1.718 sec.	898 ms (336 ms)	189 ms

Table 1: Results for a sample IPv6-to-IPv4 handover

The handover from an IPv6 to an IPv4 network depicted in the middle graph depends on various parameters as shown in table 1. The overall length of the gap is 2.81 seconds. Here is the explanation of each field in the table:

- The first delay in this table corresponds to the time required by the userland UMIP daemon to receive a link status change notification. It includes the device driver watchdog timeout, and to a greater extent, the delay before a new link-local IPv6 address is added on the interface. It is computed as the time elapsed between the last successfully transmitted data packet on the interface and the first IPv6 Router Solicitation message.

- In our implementation, IPv4 configuration is started in parallel with the IPv6 one (i.e. when the first Router Solicitation is sent) with the help of DHCP. This delay is computed as the time elapsed between the first Router Solicitation message and the first BU message using IPv4 UDP encapsulation. Among this delay, the time elapsed between the first DHCP Discover and the DHCP Ack is quoted separately for reference, but it is included in the IPv4 configuration delay.

- Finally, regular Mobile IPv6 operation is taking place using the new XFRM transformation for UDP encapsulation. This last delay is computed as the time elapsed between the first BU message and the first successfully received data packet at the correspondent node. One can notice that the BAck is received on the MR after data packets are sent on the new link. We use a mechanism called *optimistic handovers*, which allows the MN to send data packets right after sending a BU, even though it has not been acknowledged yet by the HA.

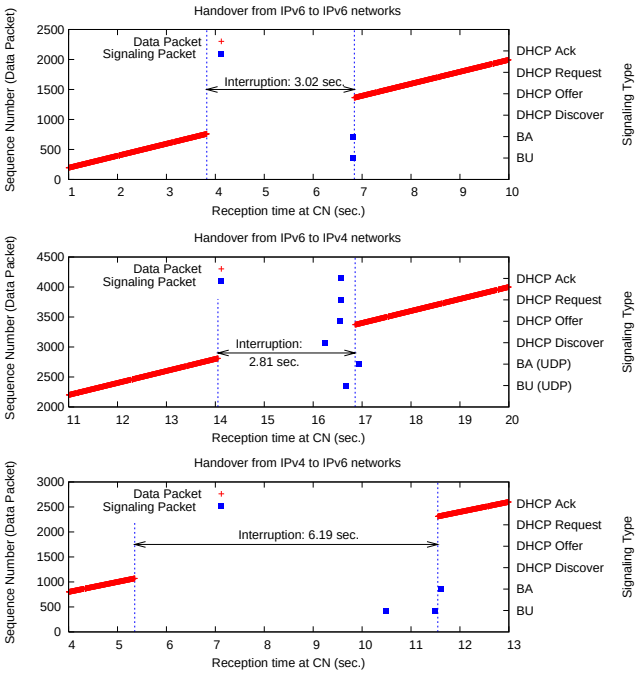

Figure 4: Handover results using a single interface

Performing the IPv4 configuration at the same time as the IPv6 one allows to greatly reduce the handover time. We consider at the moment that the IPv4 configuration takes more time than the IPv6 one. Thus, if no IPv6 CoA has been configured at the end of the IPv4 configuration, the IPv4 CoA is registered to the HA. This does not prevent though to register later an IPv6 CoA if one is available to replace the IPv4 one. Another behaviour could be to wait at least one Router Solicitation interval (4 seconds as defined in [19]) before sending a BU, but this would increase the handover time.

The handover from an IPv4 to an IPv6 network depicted in the bottom graph shows an overall gap length of almost 6.2 seconds of which the biggest part is caused by the IPv6 configuration. The link status change is barely the same as in the previous case (1.722 sec.) but the IPv6 configuration takes up to 3.42 seconds. Also, the Mobile IPv6 operations lasts 1.05 seconds due to a first BU that was actually not correctly sent on the link. We have identified several problems in the implementation that are causing these kind of performance issues. We are currently trying to figure them out.

4.2.2 *Vertical Handovers*

A second experiment was performed using a MR with two egress interfaces, each one connected to a different foreign network. The CoA of the first interface is registered to the HA. We then perform a vertical handover by disconnecting that interface, which triggers a registration update with the CoA of the second interface. As the IP configuration is already performed, the handover time is greatly reduced as shown on figure 5. Results are shown on a 30 ms window, the data packet sequence number is plotted with respect to the experiment time on the left axis. The signaling packet type is plotted with respect to the experiment time on the right axis.

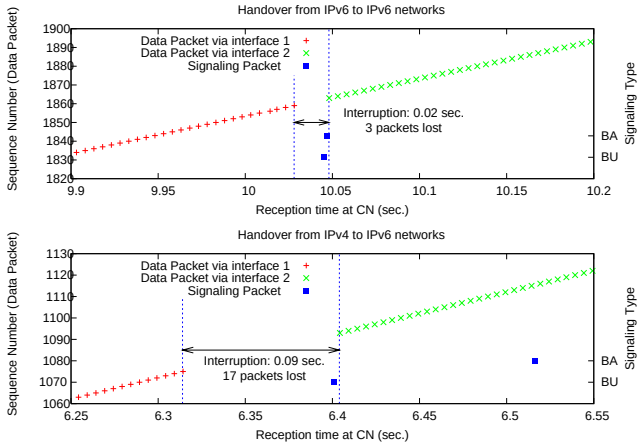

Figure 5: Handover results using two interfaces

The handover performed between two IPv6 networks and depicted on the top graph gives similar result as the one presented in a former evaluation [20]. The handover from an IPv4 to an IPv6 network depicted in the bottom graph shows a longer gap than in the IPv6-to-IPv6 handover case, which can be explained by the extra cost introduced by some of the DSMIPv6 operations (update of XFRM policies and states for the signaling, deletion and addition of different types of tunnelling interfaces). We can also see that the BU/BAck exchange is longer that in the previous case. We could notice a similar behaviour in the first experiment. This difference can be explained by the overhead introduced by DSMIPv6 on the HA (UDP decapsulation and encapsulation of the signaling messages, tunnel operations).

5. NEXT STEPS

Our implementation is still in an experimental phase. We have successfully validated the use of an IPv4 CoA, but we experienced some performance issues that we are currently addressing. The implementation also lacks several features to be compliant with the specification. The NAT detection mechanism is ready, but the NAT traversal part is still under progress: we have to use the new XFRM features to be able to encapsulate data packets in UDP. In the future, we also plan to implement the support for the IPv4 HoA that will allow to use IPv4-only applications in IPv6 networks.

In addition to that, we are also studying a possible integration with our MCoA implementation[7]. Being able to register multiple

[7]http://software.nautilus6.org/MCoA/

IPv6 and IPv4 CoAs at the same time could further reduce the handover time and improve the reliability as it has been demonstrated in IPv6-only networks in [13].

We aim at providing a stable and complete implementation of the DSMIPv6 specification for the Linux operating system. To achieve this goal, we are also planning to perform interoperability tests with other implementors. This implementation will be used in the Nautilus6 operational home agent service[8] in order to enhance the user experience by supporting the IPv4 CoAs registration.

6. SPECIFICATION ISSUES

The DSMIPv6 specification [3] being standardized at the IETF still suffers from various issues, the most important one being the IPsec integration with NAT detection and traversal. Usually to perform NAT detection, the IPv4 source address of the BU is compared by the HA with the IPv4 CoA option that was added by the MN. If they do not match, a NAT is detected on the path.

However in the case of IPsec encryption, which is mandatory for signaling messages, the IPv4 CoA option contents are only available after the payload of the new IPv6 packet has been decrypted by the IPsec stack. This usually means that the IPv4 source address of the UDP packet that contained this new IPv6 packet is no more available to the network stack, making this comparison impossible.

Although this might seem an implementation dependent consideration, it seems important to us to state that IPsec and Mobile IPv6 implementations have to be able to communicate in order to exchange this kind of information in an efficient way. In particular, some incoming packet data must traverse IPsec stack and be made available to Mobile IPv6 stack.

PF_KEY extensions [21] are a candidate solution that could allow both stacks to exchange NAT detection information, as well as tunnel information in the case where using NAT traversal and IKEv2 at the same time.

7. CONCLUSION

In this paper, we have presented a new Dual Stack Mobile IPv6 implementation on the GNU/Linux operating systems, based on the UMIP software. Use-cases of this protocol have demonstrated how the user experience could be greatly improved in an environment where the IPv4 protocol is widespread.

We have validated this implementation with a practical experiment and analyzed the results while seeking for possible enhancements. Nevertheless, some performance issues in the IPv4-to-IPv6 handover case should be quickly solved for an even better all-purpose mobility stack.

Although the system presented in this paper is already useable, some improvements can be done on several aspects as described in section 5. The combination of DSMIPv6 with the Multiple Care-of Addresses registration protocol would allow also a Mobile Node to benefit from both dual stack access networks and multi-homing, thus offering a highly reliable mobility environment to the end-user with a dual fail-safe feature.

Acknowledgments

The authors would like to thank Martin Andre and Sebastien Decugis for their work on the DSMIPv6 implementation. We also would like to thank the USAGI project members, especially Kazunori Miyazawa and Shinta Sugimoto for their precious pieces of advice during the DSMIPv6 implementation stage.

[8] http://op-ha.nautilus6.org

8. REFERENCES

[1] Vijay Devarapalli, Ryuji Wakikawa, Alexandru Petrescu, and Pascal Thubert. Network Mobility (NEMO) Basic Support Protocol. Request For Comments 3963, IETF, January 2005.

[2] Thierry Ernst and Keisuke Uehara. Connecting Automobiles to the Internet. In *ITST*, Seoul, South Korea, November 2002.

[3] Hesham Soliman. Mobile IPv6 support for dual stack Hosts and Routers. Internet Draft draft-ietf-mext-nemo-v4traversal-03, IETF, May 2008.

[4] G. Tsirtsis and H. Soliman. Problem Statement: Dual Stack Mobility. RFC 4977, IETF, August 2007.

[5] Charles E. Perkins. IP Mobility Support. RFC 3344, IETF, August 2002.

[6] David B. Johnson, Charles E. Perkins, and Jari Arkko. Mobility Support in IPv6. RFC 3775, IETF, June 2004.

[7] Massimo Bernaschi, Filippo Cacace, Antonio Pescape, and Stefano Za. Analysis and Experimentation over Heterogeneous Wireless Networks. In *TRIDENTCOM*, 2005.

[8] G. Tsirtsis and P. Srisuresh. Network Address Translation - Protocol Translation (NAT-PT). RFC 2766, February 2000.

[9] Peter J. McCann, Pat R. Calhoun, Tom Hiller, and Paal E. Engelstad. IPv6 over Mobile IPv4. Internet Draft draft-mccann-mobileip-ipv6mipv4-03, IETF, October 2002.

[10] G. Tsirtsis, V. Park, and H. Soliman. Dual Stack Mobile IPv4. Internet Draft draft-ietf-mip4-dsmipv4-06, IETF, February 2008.

[11] Changwen Liu. Support Mobile IPv6 In IPv4 Domains. In *Vehicular Technology Conference (VTC)*, May 2004.

[12] Ben McCarthy, Christopher Edwards, and Martin Dunmore. Applying NEMO to a Mountain Rescue Domain. In *WONEMO*, Japan, January 2006.

[13] Romain Kuntz and Jean Lorchat. Building Fault Tolerant Networks using a multihomed Mobile Router: a Case Study. In *AINTEC*, Bangkok, Thailand, November 2006.

[14] Romain Kuntz. Deploying reliable IPv6 temporary networks thanks to NEMO Basic Support and Multiple Care-of Addresses registration. In *WONEMO*, Japan, January 2007.

[15] Ryuji Wakikawa, Vijay Devarapalli, Thierry Ernst, and Kenichi Nagami. Multiple Care-of Addresses Registration. Internet Draft draft-ietf-monami6-multiplecoa-08, IETF, May 2008.

[16] Yoshifuji Hideaki and al. Linux IPv6 Stack Implementation based on Serialized Data State Processing. In *Special Section on Internet Technology IV, IEICE Trans Commun, Vol.E87-B, No.3*, March 2004.

[17] Bernard Aboba, James Carlson, and Stuart Cheshire. Detecting Network Attachment in IPv4 (DNAv4). RFC 4436, IETF, March 2006.

[18] Romain Kuntz, Koshiro Mitsuya, and Ryuji Wakikawa. Performance Evaluation of NEMO Basic Support Implementations. In *WONEMO*, Japan, January 2006.

[19] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor Discovery for IP Version 6 (IPv6). RFC 4861, IETF, September 2007.

[20] Jean Lorchat and Romain Kuntz. Evaluation of NEMO Communications Using Hybrid Measurement. In *ITST*, China, June 2006.

[21] S. Sugimoto, F. Dupont, and M. Nakamura. PF_KEY Extension as an Interface between MIPv6 and IPsec/IKE. Internet Draft draft-sugimoto-mip6-pfkey-migrate-04, IETF, December 2007.